



Conversational Computing, Inc. – 1493 Sandyhook, Wheaton, IL 60187  
By Charles M. Hatton  
Phone: 630-209-4472  
Application Number: 09/917,146

2    **Brief Summary of the Invention**

3

4    This patent application defines computer processes that support using the English  
5    language as a computer language. Specifically, methods are defined that let the  
6    computer: 1) process input English Language sentences from its memory which are also  
7    made up of English Language sentences. 2) use fuzzy logic to allow the computer to  
8    respond to differently worded sentences to arrive at the same action i.e., play a card  
9    game. Run solitaire. Where (play a game.) also plays the computer game of solitaire; 3)  
10    define contexts so that the computers can do: Go to New York. Who is Jim Smith? Go to  
11    Washington. Who is Jim Smith? Where answering the question, Who is Jim Smith?  
12    depends on the context: Go to New York or Go to Washington; 4) respond to multiple  
13    input sentences such as: What is the time. Get my word processor. Go to Greenburg. Who  
14    is Rod Smith; 5) where each sentence can be connected to other sentences so that in item  
15    4) Get my word processor is connected to: Go to document memory. Get word; 6)  
16    process numbers in sentences to be variables or numbers i.e. play the game of 1.222 and  
17    What is 3.44 times 22.44; 7) use multi sentence so that one sentence preceding another  
18    can change the processing methods for the successor sentence such as: show to 2 decimal  
19    places. What is 3.3333 times 2.44467? – shows answer as 8.14; 8) build the computer's  
20    memory system from English language text files; 9) talk to each other in English so they  
21    can perform parallel computing; 10) use transducers to change voice, vision, touch, other  
22    signals to English language so the application can process those signals in its English  
23    language memory systems; 11) use any device to convert to the English language; 12) do

24 inference or common sense reasoning to do the following: My car will not start. What  
25 should I do?; 13) use deductive based reasoning to do: My car will not start. What is  
26 wrong?; 14) integrate inference based reasoning with deductive based reasoning; 15)  
27 Make the computer read English language text files and store those text files in one of its  
28 memory systems; 16) make the text file that trained the application in (15) ask the  
29 application what it has learning to test that learning; 16) integrate English language data  
30 sets associated by sentence memory i.e. Go to English Works memory. Get the  
31 enhancement list. If not the enhancement list, then get design notes; 17) be a software  
32 agent to itself i.e. the said application calls another said application where the first said  
33 application is using the second said application and its specific English language memory  
34 systems to get data or cause an action or otherwise respond to the special need of the user  
35 in English; 18) conduct all correspondence to users or other devices in English as either  
36 imperative or declarative English language sentences; 19) dynamically change any or all  
37 sentence memories from external sources – take the existing sentence memories and  
38 switching them with new or different sentences memories; 20) define text and non text  
39 file memories as English language sentences; 21) make a series of sentences that are  
40 being processed to have another set of English Language sentences inserted such that the  
41 second set is a subset of the first or inserted somewhere in between the original first set of  
42 sentence and the original last set of sentences; 22) do inference based reasoning where  
43 the inference creates a new English language sentence such that this new sentence  
44 becomes new data for future inferences; 23) store English language sentences such that  
45 the same or similar sentence is stored on top of an existing sentence therefore preserving  
46 the original; 24) provide a command mode such that the said application just responds to

47 nouns; 25) communicate in English from machine to machine and human to machine  
48 and machine to human; 26) utilize the Darwinian mechanism such that sentences created  
49 by the said application, independent of those created by the user in the storage of English  
50 Language sentences in any of the said applications memories, are created by said  
51 application using inferences so that said inferences (created as English Language  
52 sentences) can be applied against the existing memory system to see if a match can be  
53 found in said memory systems; 27) build English Language sentences by machine that  
54 describe events in the computer and can be stored in text files as memory; 28) accept files  
55 to reprogram itself and or have new functionality; 29) logically unite disparate activities  
56 called from the English Language so that English Language logical analysis can occur to  
57 resolve outcomes and make decisions based on the rules of human language and  
58 governing norms; 30) oversee human language rule patterns based on other systems in  
59 order to analyze those systems and propose new solutions; 31) automatically makes a set  
60 of sentences based on one input sentence so the said application can automatically search  
61 various memories (made up of English Language sentences) to find a target memory that  
62 satisfies the intent of the original sentence; 32) read text stored in text files imported by  
63 any means (video, or any transducer input) so that said application can make judgments  
64 of said text by said application based on storage of said application memory (made up of  
65 English Language sentences); 33) learn from other devices such that the said applications  
66 memories in Figure 1 items (5), (8), and (11) are programmed in English Language  
67 sentences from outside devices either by human or machines where machines are defined  
68 as any non human device.

69

**Detailed Description of the Invention**

Description: Refer to Figure 1. The computer application known as said application (this computer software application) describes methods that allow a computer to process English Language sentences of the type: Declarative, Imperative, Interrogative, and English Language keywords. Where input (1) (refer to Figure 1 item (1)) takes in English Language text converted from 1 of N transducers. Text in ASCII format is fed to rule based parsers in Figure 1 item (2) that analyze each input sentence and parse that sentence based, in some cases, on a previous sentence. Said sentence: (What is 3.333 times 2.444?) will display the answer to 2 places when previous sentence to (What is 3.333 times 2.444?) is: (display answer to 2 places.). Said input sentence (What is 3.333 times 2.444?) finds sentence type in Math memory when user tells the said computer application to: (go to math memory.). Sequence of said sentences to compute: (What is 3.333 times 2.444?) includes the following: (Go to math memory. Display answer 2 places. What is 3.333 times 2.444?). Note: said application will process multiple sentences separated by a period, question mark or exclamation mark.

Said application is configured to open a sentence memory form (8) or (11) memories. The currently open said memory points to a Math memory as defined by a sentence in the current open memory. User (human) or machine inputs sentence: (go to math memory) and instructs the said computer application to switch to Math memory. Said Math

memory contains English Language sentences of type math such an input sentence of type: (What is 4 times 4?) matches (Number times Number) in stored memory sentence: (Number times Number is “This will multiply two numbers.”). Said sentence is a declarative sentence containing a noun and prepositional phrase (Number times Number) and a verb phrase: (is “This will multiply two numbers.”) containing the verb: is and the noun phrase contained within the double quotes: (“This will multiply two numbers”). Said sentence verb phrase contains a noun phrase defined with the double quotes even though its literal meaning is not a noun phrase. Said verb phrase is equal to: (is “”). Therefore, said equivalent sentence is: (Number times Number is “”).

Said application switches to Math memory from said input sentence: (go to math memory.) (or may be done automatically when said application analyzes input sentence and switches between English Language sentence memories), Next sentence in input sentences instructs said computer application to: (display answer to 2 places.). This sentence is found in common memory at said location (5) and implements the said instruction based on the attached action(s) to this said input sentence.

In last sentence of the above 3 input sentences: (What is 3.333 times 2.444?) matches said math memory sentence: (Number times Number is “This will multiply two number.”) and displays the said answer: (The multiplication is 8.15).

The said computer application applies input sentences from said transducers show in Figure 1 item (13) to memory systems in Figure 1 items (5), (8), and (11) looking for

116 fuzzy sentence matches which have attached actions for which any action can be an  
117 English Language sentences with is fed back into the input at Figure 1 item (13). Said  
118 application memory is made up of English Language sentences and said application can  
119 have 1 of N memories as defined by a user or copied into the said application memory  
120 system from other users at Figure 1 items (5), (8), and (11). A memory is called an  
121 object. Said input sentence is directed to 1 of N objects from a previous input sentence  
122 coming from an attached action of that sentence as in Figure 1 items (5), (8), and (11) or  
123 from the input by a user or machine as in Figure 1 item (1). Said input sentence in Figure  
124 1 item (1) or (13) where item (13) is an attached English Language sentence from said  
125 previous input sentence finds a fuzzy sentence match in current open object where said  
126 open object is selected by a previous input English Language sentence. Said open object  
127 containing English Language sentences may be matched by input sentence causing said  
128 object's sentence to execute said attached action. Said attached action can be the  
129 execution of a computer program, multiple attached English Language sentences or any  
130 program function within or outside of said application. Said computer application can  
131 send an English Language sentence in Figure 1 items (13) and (14) to another computer  
132 running said computer application. In turn, said remote computer application can  
133 respond to said computer application English Language sentence by sending back an  
134 English Language sentence to said computer application. Said remote computer  
135 application may decide to send an English Language sentence(s) to other than first said  
136 computer application. Other said computer applications running on other said computers  
137 or other devices may eventually send an English Language sentence back to original said  
138 computer application.

139

140 Said objects (memory containing English Language sentences) of said computer  
141 application can be switched by a said input sentence coming from Figure 1 item(1) or  
142 (13) such that same input can result in different actions depending on said open object.  
143 Said objects can contain same or (fuzzy similar) sentences across all objects but with  
144 different attached actions. Telling said application to: (go to your New York memory.)  
145 and asking: (Who is John Smith?) with said answer: (A person who makes shoes.) vs.  
146 telling said application to: (go to your Flordia memory.) and asking: (Who is John  
147 Smith?) with resulting said answer: (A person who lives at home.). Said application can  
148 send same input sentence (polymorphism) to said objects resulting in different results  
149 based on attachment to same sentence across all objects.

150

151 Said input sentence(s) coming from Figure 1 items (1) or (13) may be used to expose said  
152 objects (memories located in Figure 1 items (5), (8), and (11)) where said object is a class  
153 for said English Language sentences within said object . For example, directing said  
154 computer application to open its vehicle memory oject by telling said computer  
155 application to: (go to vehicle memory.) exposes the object's sentences to the next input  
156 sentence so that for example, next said input sentence coming from Figure 1 items (1) or  
157 (13) could be: (What do cars do?). Said object memory (encapsulates) details of vehicles  
158 in said object memory. Once said vehicle object memory is open by telling said  
159 computer application at Figure 1 items (1) or (13) to: (go to vehicle memory.) said next  
160 input sentence at Figure 1 items (1) or (13) is exposed to all of said sentences stored in  
161 said vehicle object memory.

162

163 Said computer application uses (data abstraction) by telling said computer application at  
164 Figure 1 items (1) and (13) to open its object memory. (where item (1) is input by human  
165 or other devices which compose an English Language sentence or item (13) where item  
166 (13) is an attached sentences stored in Figure 1 items (5), (8), and (11)). Said object  
167 memory is open by input sentence and next input sentence is exposed to said open object  
168 memory. Said open object memory contains all information on the subject of said object  
169 such that any query to said open object memory will contain information relative to the  
170 open object's memory. Said sentence: (go to my vehicle memory.) input at Figure 1  
171 items (1) or (13) exposes next said input sentence to object memories in Figure 1 items  
172 (5), (8), and (11) to a sentence like: (What has 3 wheels?). Where the open memory  
173 vehicle object containing information on all vehicles or attached sentences to other object  
174 memories can respond to a specific requests for all types of vehicles.

175

176 Said application utilizes (inheritance at the object memory level) when said input  
177 sentence to said computer application in Figure 1 items (1) and (13) causes said object  
178 memory to open in Figure 1 item (8) and (11). Said open object memory contains a said  
179 sentence with attached action that causes a common object memory to open in Figure 1  
180 item (5). Said object memory in Figure 1 items (8) and (11) inherits said corresponding  
181 common memory in Figure 1 item (5) so that a base class of sentences is combined with  
182 other object memories and their sentences shown in Figure 1 items (8) and (11). For  
183 example, said input sentence (go to your vehicle memory) in Figure 1 item (1) and (13)  
184 causes said object memory to open in Figure 1 item (8) or (11). Said open object



185 memory contains a sentence with an attached sentence (Figure 1 item (13)) to open 1 or  
186 N said common memories in Figure 1 item (5). Both object memory sentences from  
187 Figure 1 item (8) or (11) and item (5) are stored in computer RAM. User now inputs next  
188 input sentence at Figure 1 item (1) or (13) to expose open object memories Figure 1 item  
189 (5) and item (8) or Figure 1 item (5) and item (11). The vehicle open object memory in  
190 Figure 1 items (8) and (11), for example, will process input sentences at Figure 1 items  
191 (1) and (13) about vehicle attributes for that open object memory, but it will also process  
192 input sentences regarding 2 and 3 wheeled vehicles based on sentences stored in the  
193 selected common object memory. Therefore, user inputs: (go to your vehicle memory.)  
194 and said computer application opens vehicle memory and then opens the related common  
195 memory that holds sentences and actions regarding 2 and 3 wheeled vehicles. In this  
196 regard the open vehicle memory in Figure 1 item (8) or (11) inherits vehicle base class  
197 information from the corresponding common memory object in Figure 1 item (5).  
198 Typical input sentences would be as follows: (open your vehicle memory. What do cars  
199 do? What has 2 wheels?). Where: (What has 2 wheels?) comes from sentences in the  
200 common object memory (Figure 1 item (5) and opened by the sentence: (What do cars  
201 do?) located in object memories in Figure 1 items (8) and (11).

202

203 Said application utilizes (inheritance at the sentence level) when said application process  
204 English Language sentences to do inference based reasoning. Said example sentences:  
205 (My car will not start. What should I do? vs. What is wrong? For deductive based  
206 reasoning) causes said application to answer: (take a taxi or other derived answer  
207 extracted from existing said application memory). Said application analyzes input

208 sentences: (My car will not start. What should I do?) and reverses the order so that said  
209 input sentences become: (What should I do? My car will not start.) where said sentence:  
210 (What should I do?) causes common memory in Figure 1 item (5) to set up logic in  
211 Figure 1 item (2) so that said sentence: (My car will not start.) is processed using  
212 inference based reasoning. Attached action of the sentence (My car will not start.) stored  
213 in said application memory in Figure 1 items (5), (8), and (11) is (Vehicles transport  
214 things. Go to vehicle memory.) where said application goes to its vehicle memory  
215 knowing its looking for an inference from the said application based on input sentence:  
216 (What should I do?). Said application process attached sentences: (Vehicles transport  
217 things. Go to vehicle memory.) which goes to input in Figure 1 item (13) from (My car  
218 will not start.) isolating verb (transport) and applying a search in vehicle memory looking  
219 for same said verb (transport). Verb (transport) in said originating sentence with attached  
220 sentence: (Vehicles transport things.) also has a hierarchical number associated with the  
221 sentence: (Vehicles transport things.) of 1. In said target memory: (Go to vehicle  
222 memory.) searched verb (transport) at hierarchical number 1 is found in target memory  
223 with said sentence: (Cars transport people.) with attached inference: (Take a taxi.). Said  
224 application displays said inference solution to user (Take a taxi.) based on input  
225 sentences: (My car will not start. What should I do?).

226

227 The said application has many functions all of which can be implemented by inputting  
228 English Language sentences of the type declarative, imperative, interrogative, and  
229 exclamatory. To put the said application in the learn mode, a sentence like: (please  
230 learn.) and others using fuzzy logic (stored English Language sentences etc.) will store

231 English language sentences in a declarative format. (Note that fuzzy logic is defined as  
232 storing an English Language declarative (factual) sentence in 1 of N memories shown in  
233 Figure 1 in items (5), (8), and (11) such that the stored (learned sentence) defines like  
234 words by using an English Language thesaurus). All learned sentences will be stored in 1  
235 of N computer memories at locations (5), (8), and (11) in figure 1. Teaching the said  
236 application to learn how to play the computer game of solitaire can be done by storing a  
237 declarative sentence like: (card game of solitaire is "Playing a solitaire card game.") and  
238 attaching the solitaire program executable (sol.exe) to the said declarative sentence. Any  
239 number of attached actions can be stored with a learned declarative English Language  
240 sentence including other English Language sentences that are fed back into the input at  
241 Figure 1 item (13) or to the output item (14). From the stored sentence: (card game of  
242 solitaire is "Playing a solitaire card game.") any declarative, imperative, interrogative,  
243 and exclamatory sentence from machine to human can input an ASCII text sentence to  
244 play a solitaire card game using said application. Some of the word combinations  
245 include: (please play cards, get solitaire, play a game, load a card game, Can you play  
246 solitaire?) etc. These word combinations are formed using English Language sentence  
247 rules and implemented by humans or machines and input into said application in Figure  
248 1 item (1) or item (13) to match words in object memory in Figure 1 items (5), (8), and  
249 (11). Said input sentences finds object memory sentence in Figure 1 items (5), (8), and  
250 (11) which has been learned and stored by user in declarative English Language format as  
251 in: (card game of solitaire is "Playing a solitaire card game."). Each of the input  
252 sentences (declarative, imperative, interrogative, and exclamatory) will match various  
253 words of the learned/stored declarative English Language sentence to cause said

254 computer application to play the computer game of solitaire. The method described  
255 above in Figure 1 item (5) and (8) learns declarative English Language sentences in  
256 Native mode and stores those sentences in 1 of N user object memories stored in  
257 computer files. File memories are defined by users and contain English Language  
258 declarative sentences and associated stored actions as defined by the user. A stored  
259 action is any event or another English Language sentence that may be attached to a stored  
260 declarative sentence. Said stored actions can instruct said application to switch to a new  
261 object memory file for which the next input English Language sentence from said  
262 transducers in Figure 1 item (1) can cause new functionality to occur. Likewise said  
263 application can use text files (shown in Figure 1 item (11)) that are English Language  
264 declarative sentences separated into paragraphs as shown in Figure 2.

265

266 Figure 2 (Items 1 – 22)

267

268 1. play a card solitaire game.

269 do: (sol.exe, c:\Windows).

270 display: Playing the card game of solitaire.

271

272 2. play a board chess game.

273 do: (chess.exe, c:\Windows).

274 display: Looking at my Files.

275

276 3. get my your this text paragraph file.

277 do: (notepad.exe, \Computer text memory\game text memory.txt).

278

279 4. who is Bill William Jefferson Clinton?

280 do: play cards.

281 do: show my computer files.

282 display: A guy who lives in New York.

283 related: His mother lives in Colorado.

284

285 5. who is Hillary Hilary Rodam Clinton?

286 display: Wife of the President of the United States.

287 related: Hillary's parents.

288 do: search. play cards.

289

290 6. get me picture map of England Great Britain.

291 do: (president.bmp, c:\MyPro386w\bitmap files\uk.bmp).

292 display: Showing a map of England.

293

294 7. find a chair car or boat.

295 do: play cards. go to core.

296

297 8. show my C disk drive space availability capacity amount on my C drive.

298 do: (Drvspace.exe, c:\).

299 display: Showing the c drive space.

300

301

302 8a. show my A disk drive space availability capacity amount on my C drive.

303 do: (Drvspace.exe, a:\).

304 display: Showing the a drive space.

305

306 8b. do disk clean or defragmentation.

307 do: (defrag.exe, c:\Windows).

308 display: defrag the selected disk drive.

309

310 9. who owns or is the owner of the Chicago and Northwestern?

311 do: play solitaire.

312

313 10. play a chess board game.

314 display: playing chess.

315

316 11. call a Person.

317 do: If get Person's phone number. then call Person. else call information.

318 display: sentence logic test. Also, Chuck's phone number should be asserted into

319 memory which would allow call Chuck to happen.

320

321 12. call 411 information.

322 do: (sol.exe, c:\Windows).

323

324 12a. what does Intel PC notes makes computers easy to use say.

325 do: (Intel ease of use.bmp, c:\MyPro386w\bitmap files\Intel ease of use.bmp).

326 display: From Intel's web site on 12/15/98.

327

328 13. He was in New York getting his car.

329 display: getting his car in New York.

330

331 14. This is a Chuck test.

332 display: This is a Chuck test.

333

334 15. get calculator adder my math machine.

335 do: (calc.exe, c:\Windows).

336 display: can also be made into a scientific calculator.

337

338 16. something destroyed the bark.

339 do: play cards.

340

341 17. show my design note additions.

342 do: (notepad.exe, a:\other\design notes1).

343 display: These are the design notes on the ews diskette.

344

345 18. show the my problems issues list.

346       do: (notepad.exe, a:\Other\ews problem list).

347       display: Ews problem list.

348

349   19. go get the other next alternative Bill.

350       do: get test 1. Who is Bill. get test 2.

351       display: Getting information on the other Bill.

352

353   20. something destroyed the bark.

354       do: play cards.

355

356   21. get the new proposed functions functional capabilities notes list.

357       do: (notepad.exe, a:\Other\function list).

358

359   22. set up Ews development environment platforms software tools.

360       do: do Dos. show problem list. show functional notes. show design notes.

361

362   End of Figure 2 (Items 1 – 22)

363

364   Said paragraphs of English Language declarative sentences can be matched by an input

365   English Language sentence from machine to human of type declarative, imperative,

366   interrogative, and exclamatory. Matching the input English Language sentence of type

367   declarative, imperative, interrogative, and exclamatory with English Language

368   declarative sentences stored in each text paragraph as shown in Figure 2 such that one



369 paragraph with attached actions for which one action can be made up of an English  
370 Language sentence will call another paragraph sentence who's attached action could call  
371 Native memory which switches to a new text file shown in Figure 1 using mechanisms in  
372 items (5), (8), and (11). Text file memory switching will enable the following: (Go to  
373 Washington, D.C. What is Mr. Smith's phone number? Go to New York. What is Mr.  
374 Smith's phone number?). Where the sentences (Go to Washington, D.C. and Go to New  
375 York.) switch memories giving the said application through parsers in Figure 1 item (2)  
376 the ability to point to a new set of English Language memories defined in Figure 1 items  
377 (5), (8), and (11).  
378  
379 Said application can read a text files made of up of English Language sentences found by  
380 telling said computer application to go out and read said text file such that read text file  
381 teaches said application by transferring English Language sentences from said text files  
382 into said application in Figure 1 item (1). Once said application has transferred English  
383 Language sentences into said application memory object in Figure 1 item (8), said  
384 application continues to read said text file which further instructs said application, in  
385 English Language, to do what said computer application has learned by storing English  
386 Language sentences in said computer application object memory in Figure 1 item (8).  
387 Said text file that teaches said computer application is shown in Figure 3. Said  
388 application is told, using an English Language sentence in Figure 1 item (1) or (13) to  
389 learn from a text file when user instructs said computer application to go out and read  
390 said text file. The said text file that teaches the said computer application to play the  
391 computer game of solitaire is as follows:

392

393 Figure 3

394

395 Go to your learning memory. Save this data. "sol exe".

396 Save the user sentence. "solitaire card game" is "playing solitaire".

397 Stop learning text sentences. play solitaire. Go to your user memory.

398

399 End of Figure 3

400

401 Said common memory in Figure 1 item (5) contains English Language declarative  
402 sentences stored in ASCII text such that stored English Language sentence in common  
403 memory is matched with an input English Language input sentence of the form  
404 declarative, imperative, interrogative, and exclamatory located in Figure 1 item(1) or (13)  
405 from machine or human. When said match is made in common object memory with  
406 input sentence, search of subsequent memories in Figure 1 items (8) and (11) is  
407 prevented. If search of common memory fails to find a stored English Language  
408 sentence that matches input sentence using fuzzy logic, search continues in Figure 1 item  
409 (8) and then to item (11). Said application in Native memory in Figure 1 item (8) or item  
410 (11) can have a stored English Language sentence attached to a stored action that  
411 switches common memory in Figure 1 item (6). Said common memory switches to a  
412 new set of English Language sentences which may become aligned to the context of said  
413 Native memory in Figure 1 item (8) and item (11) as defined by the user when said

414 application is in learn mode storing actions with said memories in Figure 1 item (8) and  
415 (11).

416

417 Said common memory in Figure 1 item (5) enables the following functionality to occur in  
418 said application using deductive and inference based reasoning. In deductive based  
419 reasoning input English Language sentences show in Figure 1 item (1) may include a  
420 stream of declarative sentences such as: (My car will not start. There are no headlights.  
421 What is wrong?). Where said sentence: (What is wrong?) resides in common memory and  
422 causes deductive based processing of said two input sentences: (My car will not start.  
423 There are no headlights.). Said common memory sentence stored in Figure 1 item (5)  
424 uses fuzzy logic such that variation of said sentence: (What is wrong?) can include: (get a  
425 solution. What is it?) where these variations are connected to the same action causing  
426 said deductive solution to the English Language input sentences in Figure 1 item (1): (My  
427 car will not start. There are no headlights.). Said common memory sentences (What is  
428 wrong? etc.) have stored actions attached to said sentences as shown in Figure 1 item (5)  
429 causing the appropriate process to occur when said sentences (My car will not start.  
430 There are no headlights.) are processed in memories in Figure 1 items (8) and (11).

431

432 Said declarative input sentences for deductive based reasoning are fired into computer  
433 memory based on matches found in Native and text memory at locations Figure 1, items  
434 (8) and (11). Stored actions associated with each declarative input sentence define a  
435 confidence factor when all input declarative input sentences match all stored declarative  
436 sentences shown in Figure 1 items (8) and (11). Associated stored actions found with

437 matching input sentences and stored declarative sentences found in items (8) and (11)  
438 cause an expected result which could include sending a new English Language sentence  
439 to the input located in Figure 1 item (13). The stored actions may contain a number of  
440 sentences the are either fed back to the input in at Figure 1 item (13) or to external  
441 computers, humans, or appliances in Figure 1 item (14). Where an appliance accepts an  
442 English Language sentence, reacts to that sentence, and sends an English Language  
443 sentence back to the sending device as in Figure 1 item (1) or other devices in a network.  
444 All generated English Language sentences resulting from memory transactions whether  
445 generated from a stored action or composed using internal parsing technology in Figure 1  
446 item (2) interact with the memory of said computer application or the memories of other  
447 devices creating a dialog using English Language sentences between devices. Any  
448 decoded English Language sentence results in the activation of a stored action. A simple  
449 device may decode specific English Language sentences (decoded to ASCII text for  
450 processing) sent from said application resulting in said device sending back an English  
451 Language sentence in response to said application or other said applications or other  
452 devices or humans either as text, voice or other electrical signals represented by said  
453 English Language sentence(s). Said stored actions result in new sentences or memory  
454 switching as shown in Figure 1 items (4), (6), (7), (9), and (10) from said application or  
455 canned sentences from appliances causing said work to be done for user.

456

457 When said application is not in the learn mode, English Language declarative input  
458 sentences shown in Figure 1 item (1) find stored declarative sentences in Figure 1 item  
459 (8) and item (11). When found, those declarative sentences are connected to stored

460 actions which define the solution for deductive based reasoning. Associated solutions  
461 include English Language sentences that may be fed back to the input Figure 1 item (13)  
462 or go outside of the said computer application shown at Figure 1 item (14) to other  
463 devices. Those devices can be said computer application running on other computers or  
464 appliances. In all cases, outside sources, either human or machine, communicate to each  
465 other in English Language sentences.

466

467 Said computer application can integrate deductive based reasoning with inference based  
468 reasoning. Deductive based reasoning is the storage of English Language sentences  
469 connected together within and across object memories. Connecting sentences and object  
470 memory together occurs in the learn mode as user defines connecting relationships  
471 between sentences. User typically enters declarative sentences in Figure 1 items (1) and  
472 (13) which map into learned stored connected sentences in object memory in Figure 1  
473 items (5), (8), and (11). A typically example session could include the following input  
474 sentences: (go to vehicle memory. My car will not start. There is no dome light. What is  
475 wrong?). When said computer application receives first said sentence: (go to vehicle  
476 memory.) said application switches to vehicle object memory in Figure 1 items (8) and  
477 (11). Once switched, said second sentence: (My car will not start.) finds a sentence  
478 match in current open object vehicle memory. Matched said sentence (My car will not  
479 start.) has an attached sentence: (go to start problem memory.) Said computer application  
480 builds new remaining sentences: (go to start problem memory. There is no dome light.  
481 What is wrong?) and said computer application switches to: (go to start problem  
482 memory) in Figure 1 item (13) causing new object memory to open in Figure 1 items (8)

483 and (11). Next input sentence of remaining sentences: (There is no dome light. What is  
484 wrong?) finds matching sentence in current open object memory (go to start problem  
485 memory.) in Figure 1 items (8) and (11). As said matching sentences are found  
486 throughout object memories, said attached actions to each matching sentence is stored in  
487 said computer application RAM area noting the number of matched sentences compared  
488 to the number of matched sentence with a group as defined by the user in said application  
489 learn mode. Said last sentence: (What is wrong?) causes said application to find said  
490 sentence in object common memory in Figure 1 item (5). Said sentence: (What is  
491 wrong?) has attached actions that causes said application to process data stored in said  
492 computer application RAM resulting form said sentence matches found in said object  
493 memory in Figure 1 items (8) and (11). Resulting said solution: (The car battery may be  
494 dead or disconnected. Confidence factor 100 percent). Said data in Ram may also  
495 include sentences: (go to battery memory. battery provides power.) such that when user  
496 inputs: (What should I do? – asking said application to make an inference or alternative  
497 to using a battery, and where said sentence: (What should I do?) exists in said common  
498 object memory in Figure 1 item (5)) said application transfers said sentences: (go to  
499 battery memory. Battery provides power.) to Figure 1 item (13). Said inference  
500 sentences: (go to battery memory. Battery provides power.) switch to battery memory as  
501 directed by input sentence: (go to battery memory.) in Figure 1 items (8) and (11). Next  
502 said input sentence: (Battery provides power.) finds said sentence in battery object  
503 memory in Figure 1 items (8) and (11). Said sentence: (Battery provides power.) has  
504 attached sentence: (go to electrical power memory.) causing said application to switch  
505 object memory in Figure 1 items (8) and (11). Said application isolates verb (provides)

506 and searches through object memory: (go to electrical power memory.) for said verb  
507 (provides) at same said hierarchical level of 2 as defined by said previous sentence:  
508 (Battery provides power.). Said computer application finds sentence in said open object  
509 memory (go to electrical power memory.) in Figure 1 items (8) and (11) finds sentence:  
510 (Devices provide electrical power.). Said found sentence: (Devices provide electrical  
511 power.) at hierarchical level 2 has attached user comment: (Replace or recharge battery.).  
512  
513 Said application uses multi sentence technology to: 1) open new object memory; 2) find  
514 matches in object memory in Figure 1 items (5), (8), and (11) as defined by input from  
515 Figure 1 items (1) and (13); 3) cause said found sentence to execute any action as  
516 defined by said attached computer programs; 4) send attached sentences found in  
517 memory at Figure 1 items (5), (8), and (11) to output in Figure 1 items (14) to said other  
518 computers using said application or devices responding to said English Language  
519 sentences or their said symbolic derivatives; 5) enable parallel computing where said  
520 computer or human dialog in English Language is sustained using said application within  
521 or across computer platforms is complete and dialog stops such that all sentences have  
522 competed resulting in work being done by said application. Said found sentences in  
523 Figure 1 items (5), (8), and (11) apply new sentences to input at Figure 1 item (13). Input  
524 sentences at Figure 1 item (1) or item (13) can be from 1 to N sentences where sentences  
525 from Figure 1 item (13) can be attached to stored sentences in Figure 1 items (5), (8), and  
526 (11). Input sentences in Figure 1 item (1) can be composed by human or machine in any  
527 number from 1 to N.  
528

529 Said application logically processes input sentences such that the following example  
530 input sentences coming from Figure 1 item (1) or (13) from deposited sentences in  
531 Figure 1 items (5), (8), and (11) are: (please play cards. If you played cards then show  
532 card instructions. Otherwise, play a game of chess.) Where said application finds match  
533 in said object memory in Figure 1 items (5), (8) and (11) for said sentence: (please play  
534 cards.) noting that if said match did not occur from input in Figure 1 items (1) or (13) and  
535 said open object memory in Figure 1 items (5), (8), and (11) that said application  
536 generates internal message: (my memory is blank.). When said message is generated (my  
537 memory is blank.) because said application could not find match to said input sentence:  
538 (please play cards.) to said memory, said next input sentence: (If you played cards then  
539 show card instructions.) is disregarded such that the sentence: (Otherwise, play a game  
540 of chess.) is processed through said application memory in Figure 1 items (5), (8) and  
541 (11) resulting said application playing the computer game of chess. Said application with  
542 said stored English Language sentences stored in said object memories in Figure 1 items  
543 (5), (8), (11) have attached sentences that can be sent to said application output in Figure  
544 1 item (14) or display on said application user screen or put into ASCII text to voice  
545 transducer such that when said input sentences matches said memory sentences with said  
546 attached action which could be a said English Language sentences, that said sentence  
547 could be announced to a local or remote speaker.

548

549 Said application can be programmed in ordinary English Language sentences by storing  
550 said sentences in said object memories in Figure 1 items (5), (8), and (11). Format of  
551 said stored English Language sentences can be generated by computer program and auto



552 loaded into said application object memory. Said application can request auto load of  
553 new said application object memories to adapt to new requirements as defined by an  
554 English Language sentence form human or machine.

555

556 Said application uses two methods to sort through user object memory in Figure 1 items  
557 (5), (8), and (11). Method 1 indexes said object memory in Figure 1 item (8) to open  
558 object memories in Figure 1 item (11) with single sentence input or multi sentence input.  
559 Method 2 builds internal sentences to reapply those sentences in Figure 1 item (13) in  
560 order to sort through object memory in Figure 1 items (5), (8), and (11). In Method 1,  
561 human or machine enters sentence(s) in Figure 1 item (1) or said application enters  
562 sentence(s) in Figure 1 item (13). Said object memory switches object memory based on  
563 attached sentence associated with matched sentence in Figure 1 items (5), (8), and (11).  
564 For example, the input sentence(s): (Go to New York memory. Who is John Smith?)  
565 causes said index sentence: (Go to New York memory.) to switch to New York object  
566 memory when said application is told with an English Language sentence to: (please go  
567 into search mode.). (Go to New York memory.) is in an index of all object memories  
568 located in Figure 1 item (8). Attached action of said found sentence (Go to New York  
569 memory.) switches object memory to New York memory in Figure 1 items (8) or item  
570 (11). Said next sentence locates said sentence (Who is John Smith?) through existing  
571 index or in memories in Figure 1 items (11) or items (5). Said open object memory is  
572 open containing 1 of N sentences of which one is fuzzy similar to (Who is John Smith?).  
573 Said next sentence: (Who is John Smith?) searches for a match in said open object  
574 memory in Figure 1 items (8), and (11) and displays answer to (Who is John Smith?) as

575 defined in said computer application learn mode. In said learn mode, user would have  
576 previously stored and defined a sentence like: (“John Smith New York friend” is “John  
577 Smith is a friend from New York.”) whereby the input sentence: (Who is John Smith?)  
578 in Figure 1 items (1) or (13) would look for a match in Figure 1 items (5), (8), and (11)  
579 such that the input sentence is parsed (broken apart according to the rules: sentence =  
580 noun phrase + verb phrase) in Figure 1 item (2) or their derivatives and applied against  
581 the stored sentence: (Who is John Smith?) found in Figure 1 items (5), (8), and (11).  
582 Found said sentence: (Who is John Smith?) may have 1 of N stored actions. Said stored  
583 actions can include any combination of English Language sentences and other data types  
584 depending on user configuration of stored sentence in Figure 1 items (5), (8), and (11).  
585 Said stored actions are attached to said stored sentence in said computer application using  
586 said computer application learn mode. Said stored sentences and said stored actions are  
587 stored in Native, text or SQL data types as shown in Figure 1 items (5), (8), and (11).